**VŠB-Technická univerzita Ostrava**
**Fakulta elektrotechniky a informatiky**
**Katedra informatiky**

# Analýza cen v agregátorech zboží z pohledu prodejce

# Price Analysis for Shop Gods Agragator from the Reseler Poin on View

2012 Jiří Sedláček

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

# Zadání bakalářské práce

Student: **Jiří Sedláček**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Analýza cen v agregátorech zboží z pohledu prodejce
Price Analysis for Shop Gods Agragator from the Reseler Poin on View

Zásady pro vypracování:

1. Nastudujte problematiku agregátků zboží na internetu (Zbozi.cz, Heureka.cz, Jyxo.cz), definujte jejich možnosti, výhody a nedostatky.
2. Zaměřte se na problematiku publikování prodejních nabídek v těchto agregátorech.
3. Na základě vybraného elektronického obchodu, který publikuje svou nabídku v agregátorech, proveďte analýzu možností získávání strategických obchodních dat z těchto zdrojů.
4. Cílem práce je implementace nástroje, který by pomohl zadavateli analyzovat vlastní nabídky v agregátorech a porovnat je s konkurencí. Na základě toho definovat doporučení ke změně ceny produktu.
5. Proveďte analýzu, návrh a implementaci této problematiky, které bude finálně začleněna do již existujícího řešení pro SEO optimalizace.
6. Zhodnoťte dosažené výsledky a další možnosti rozšíření.

Seznam doporučené odborné literatury:

[1] KUBÍČEK, Michal. Velký průvodce SEO : Jak dosáhnout nejlepších pozic ve vyhledávačích. Vydání první. Brno : Computer Press a. s., 2008. 318 s. ISBN 978-80-251-2195-5.
[2] KUBÍČEK, Michal; LINHART, Jan. 333 tipů a triků pro SEO. Vydání první. Brno : Computer Press a. s., 2010. 262 s. ISBN 978-80-251-2468-0.
[3] GRAPPONE, Jennifer; COUZIN, Gradiva. SEO - Search Engine Optimalization. Překlad: Roman Skřivánek, Dana Balaštíková. Vydání první. Brno : ZONER software, s.r.o., 2007. 328 s. ISBN 978-80-86815-85-5.
[4] JANOUCH, Viktor. Internetový marketing : Prosaďte se na webu a sociálních sítích. Vydání první. Brno : Computer Press a. s., 2010. 304 s. EAN: 9788025127957.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Radoslav Fasuga, Ph.D.**

Datum zadání: 18.11.2011

Datum odevzdání: 04.05.2012


doc. Dr. Ing. Eduard Sojka
*vedoucí katedry*

prof. RNDr. Václav Snášel, CSc.
*děkan fakulty*

# Prohlášení o autorství

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne 4.5.2012

Podpis:

# Poděkování

Rád bych na tomto místě poděkoval svému vedoucímu Ing. Radoslavu Fasugovi, Ph.D. za jeho velmi odborné vedení, cenné rady a připomínky.

# Abstrakt

Bakalářská práce se zabývá návrhem agregátoru zboží a jeho implementací. Dále popisuje postupy pro párování produktů a srovnávání jejich cen. Obsahuje popis problémů ukládání dat, a vyhledávání v těchto datech. V teoretické části této bakalářské práce se zabývám nejznámějšími českými agregátory zboží a popisem jejich služeb. Dále je v této části možné nalézt popisy technologických řešení využitých pro implementaci softwarového řešení. V části druhé se zabývám analýzou, návrhem, popisem algoritmů, kterými řeším párování produktů a srovnávání cen. Také je zde obsažen popis možností systému a zhodnocení výsledků.

# Klíčová slova

agregátor, zboží, fulltext, PHP, MySQL, cena

# Abstract

This thesis describes design of product aggregator and his implementations. It also describes procedures for product matching and comparing their prices. It contains description of the problems of data storage and search in these data. In the theoretical part of this thesis deals with the most famous Czech product aggregators and description of their services. In this section can be found descriptions of technological solutions utilized for the implementation of software solutions. The second part deals with the analysis, design and description of algorithms, which solve the pairing of products and price comparisons. Also included is a description of the capabilities of the system and the results.

# Keywords

aggregator, products, fulltext, PHP, MySQL, price

# Seznam obsažených obrázků a tabulek

Obrázek 1: Logo Zboží.cz ..... 2

Obrázek 2: Logo Heureka.cz ..... 3

Obrázek 3: Logo Nákupy Google ..... 4

Obrázek 4: Příklad xml feedu pro Nákupy Google ..... 4

Obrázek 5: Logo PHP ..... 6

Obrázek 6: Logo MySQL ..... 8

Obrázek 7: Přehled tabulek systému ..... 16

Obrázek 8: Use case diagram ..... 17

Obrázek 9: Vývojový diagram ..... 18

Obrázek 10: Úvodní stránka ..... 20

Obrázek 11: Aktualizace dat ..... 23

Obrázek 12: Detail produktu po vyhledání ..... 24

Obrázek 13: Výsledky vyhledávání ..... 25

Obrázek 14: Návrhy na párování produktů ..... 27

Obrázek 15: Grafické znázornění časových režií aktualizace ..... 32

Obrázek 16: Generování cache ..... 34

Obrázek 17: Vrácení výsledků z cache ..... 35

Tabulka 1: Ukázka datového slovníku ..... 15

Tabulka 2: Časové přehledy funkcí ..... 31

Tabulka 3: Statistika nahrávání dat ..... 33

# Seznam použitých zkratek a symbolů

| | |
|---|---|
| **SQL** | Standardizovaný dotazovací jazyk |
| **MySQL, PostgreSQL, Oracle, ODBC, MSSQL** | Různé SŘBD |
| **PHP** | Skriptovací programovací jazyk |
| **SŘBD** | Systém řízení báze dat |
| **XML** | Obecný značkovací jazyk |
| **EAN-13** | Typ čárového kódu |
| **URL** | Jednotný lokátor zdrojů – určuje umístění informace |
| **W3C** | Konsorcium vyvíjející webové standardy |
| **(X)HTML** | (Rozšiřitelný) značkovací jazyk pro webové dokumenty |
| **WML** | Značkovací jazyk pro mobilní webové dokumenty |
| **HTTP** | Protokol pro výměnu webových dokumentů |
| **TCP/IP** | Sada protokolů pro komunikaci v internetu |
| **InnoDB, MyISAM** | Úložiště v MySQL |
| **Feed** | Zdroj dat |
| **Apache** | Webový server |
| **PhpMyAdmin, Adminer** | Nástroje pro správu databáze |
| **PSPad** | Editor zdrojových kódů |

# Obsah

1 Úvod ..... 1

2 Agregátory ..... 2

2.1 Agregátory zboží ..... 2

2.1.1 Zboží.cz ..... 2

2.1.2 Heureka.cz ..... 3

2.1.3 Nákupy Google ..... 3

2.1.4 Další agregátory ..... 4

2.2 Agregátory slev ..... 5

3 Využité technologie a standarty ..... 6

3.1 XML ..... 6

3.2 PHP ..... 6

3.3 MySQL ..... 8

3.4 EAN-13 ..... 9

4 Možnosti systému - funkční požadavky, technické požadavky ..... 10

4.1 Uživatelské role ..... 10

4.2 Klíčové funkce ..... 10

4.2.1 Uživatelská sezení ..... 10

4.2.2 Agregátor ..... 11

4.2.3 Nastavení uživatele ..... 11

4.2.4 Administrace ..... 11

4.2.5 Párování produktů ..... 11

4.2.6 Doporučení změny ceny produktu ..... 12

4.3 Vstupy do systému ..... 12
4.4 Výstupy ze systému ..... 12
4.5 Okolí systému ..... 13
4.6 Technické požadavky ..... 13
5 Analýza a návrh ..... 14
5.1 Datová analýza ..... 14
5.1.1 Lineární zápis entit ..... 14
5.1.2 Datový slovník ..... 15
5.1.3 Konceptuální model ..... 16
5.2 Funkční analýza ..... 17
5.2.1 Use case diagram ..... 17
5.2.2 Procesy a funkce ..... 18
6 Implementace a testování ..... 19
6.1 Volba platformy a úložiště dat ..... 19
6.2 Běhové prostředí a vývojové nástroje ..... 19
6.3 Implementace ..... 20
6.4 Zjednodušené popisy zajímavých algoritmů ..... 21
6.4.1 Testování uživatelských feedů ..... 21
6.4.2 Popis každodenní aktualizace dat pro vyhledávání: ..... 21
6.4.3 Nastavení MySQL pro fulltextové vyhledávání: ..... 23
6.4.4 Fulltextové vyhledávání ..... 24
6.4.5 Vyhledávací dotaz ..... 25
6.4.6 Změna ceny produktu: ..... 26
6.4.7 Párování produktů ..... 26

6.4.8 Metody párování produktů ..... 27
7 Časové a výpočetní náklady systému ..... 30
7.1 Nejnáročnější procesy systému ..... 30
7.1.1 Každodenní aktualizace dat ..... 30
7.1.2 Předpokládaná časová náročnost reálně nasazeného systému ..... 32
7.1.3 Statistika nahrávání dat do systému ..... 32
7.1.4 Faktory ovlivňující rychlost aktualizace ..... 33
8 Výhody ukládání uživatelských dotazů ..... 34
8.1 Předgenerování výsledků do cache ..... 34
8.1.1 Postup generování ..... 34
8.1.2 Postup vrácení výsledku ..... 35
8.2 Opravy překlepů ..... 35
9 Zhodnocení výsledků a srovnání s konkurencí ..... 36
10 Závěr ..... 37
Seznam použité literatury ..... 38
Přílohy ..... 39

# 1 Úvod

Agregátor - slovo, které v poslední době proletělo českým internetovým polem jako blesk. Agregátor je slovo vzniklé ze slova agregát, které podle slovníku cizích slov znamená spojení, propojení, seskupení. Tedy agregátor je něco (systém), co spojuje, propojuje, seskupuje. V našem případě se jedná o systém, který seskupuje zboží, popřípadě slevy.

Setkáváme se s agregátory zboží, s agregátory slev, s agregátory agregátorů. Služby nejrůznějších agregátorů se naučili velmi rychle využívat snad všichni uživatelé internetu. Tito uživatelé by se dali roztřídit do dvou skupin: na ty, kteří nabídky v agregátorech publikují a na ty, kteří tyto nabídky čtou a následně využívají. Existuje však i třetí strana, strana provozovatele agregátoru a lidí starajících se o jeho chod.

Dnešní náročný uživatel - zákazník hledá co nejnižší cenu a zároveň také určitou solventnost prodejce. Má také celou řadu dalších požadavků, kterým se provozovatelé agregátorů snaží vyjít vstříc a nabízí nejrůznější doplňkové funkce, jako jsou hodnocení prodejce, řazení výsledků vyhledávání podle rozličných kritérií. Avšak nesmíme zapomínat na prodejce, jehož cílem je prodat zboží. Může se snažit získat zákazníka nižší cenou, svojí solventností - tedy svým dobrým hodnocením v agregátoru, nebo placenou reklamou tamtéž.

Hlavním cílem této práce je vývoj systému, který nabídne standartní agregátor zboží pro zákazníka a systém publikování nabídek prostřednictvím agregátoru zákazníkovi s doporučením prodejní ceny pro prodávajícího. Z toho důvodu bylo nutné nastudování chování a funkčností tuzemských agregátorů zboží.

# 2 Agregátory

## 2.1 Agregátory zboží

Agregátor zboží je služba běžící v prostředí internetu. Tato služba umožňuje svým uživatelům vyhledávat zboží, které má uložené v databázi na základě jejich požadavku a toto třídit podle zadaných kritérií. Informace o zboží těmto službám poskytují samotné internetové obchody, které si tak samy sobě dělají reklamu, informují, že toto zboží prodávají atd. Data od internetových obchodů si agregátory stahují automaticky a to většinou ve formátu xml, který má přísně daná specifika, jejichž nedodržení může znamenat vyloučení ze zobrazování produktů toho konkrétního internetového obchodu. Agregátory zboží jako takové zboží neprodávají, ale pouze zprostředkují jeho prodej. Prodej zboží si internetové obchody zajišťují samy.

Agregátory zboží jsou schopny vyhledávat podle nejrůznějších kritérií. Hlavním je vyhledávání podle názvu a popisu zboží. Dále se tyto výsledky dotřiďují například cenovým intervalem, dostupností, způsobem platby, umístěním obchodu. Tyto výsledky je nadále možné třídit podle ceny - od nejnižší po nejvyšší a podle toho jestli je zboží nové, nebo použité.

### 2.1.1 Zboží.cz

Zboží.cz je internetová služba - agregátor zboží, která je schopna vyhledávat informace o zboží nabízeném prostřednictvím internetových obchodů - tzv. e-shopů.

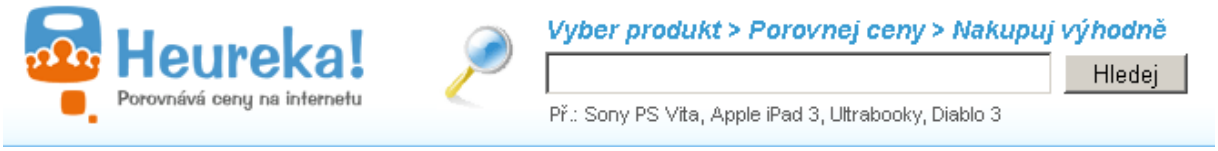

Obrázek 1: Logo Zboží.cz

Zboží.cz má vytvořené kategorie zboží, kterými mohou návštěvníci procházet a zboží si v dané kategorii najít a zobrazit kartu produktu se základním popisem a přehled e-shopů, které mají tento produkt v nabídce.

Další možností jak se k hledanému zboží dostat je použít vyhledávač, kam uživatel zadá název zboží a po odeslání formuláře, jsou mu předloženy výsledky. Pokud uživatel zadá do vyhledávače název zboží, které je takzvaně napárované na stejné produkty, zobrazí se rovnou odkaz na kartu zboží. V případě, že tento agregátor nemá na vyhledávaný dotaz napárované stejné produkty, zobrazí se výčet produktů nalezených fulltextovým vyhledávačem. Seznam

### 2.1.2 Heureka.cz

Heureka.cz je dalším z českých agregátorů zboží. Mezi její přednosti patří zejména uživatelské hodnocení internetového obchodu po nákupu zboží a v kartě zboží záložky specifikace, recenze, poradna.




Obrázek 2: Logo Heureka.cz

Mimo samotné porovnávání cen zboží nabízí detailní zobrazení parametrů produktu společně s popisem pod záložkou specifikace. Uživatelské recenze a komentáře je možné nalézt v záložce recenze. Pod záložkou poradna, mohou návštěvníci systému vznášet dotazy ohledně zboží. Na tyto dotazy odpovídají jak další uživatelé systému heuréka.cz, tak prodejci, nabízející toto zboží. Velice zajímavou funkcí toho agregátoru je bezpochyby Hlídání ceny produktu. Uživatel systému si může u konkrétního zboží nastavit cenu, pod kterou když klesne cena tohoto zboží, tak je o této skutečnosti informován emailem. Za zmínku stojí také spolupráce s aukčním systémem Aukro.cz

### 2.1.3 Nákupy Google

Nákupy google jsou službou společnosti Google, která poskytuje agregátor zboží.


Obrázek 3: Logo Nákupy Google

Zboží je řazeno do kategorií a přes ně je možné dostat se na kartu produktu. Obdobně jako u heuréky je zde možné nalézt specifikace produktu, seznam obchodů, které tento produkt nabízejí a ceny v těchto obchodech.

Tento agregátor má jiná specifika co se týče zdrojových xml feedů než tuzemské agregátory. [1][2][3]

```
<?xml version="1.0"?>
<rss version="2.0"
xmlns:g="http://base.google.com/ns/1.0">
<channel>
<title>Název zdroje dat</title>
<link>http://www.example.com</link>
<description>Popis obsahu</description>
<item>
<title>Červený vlněný svetr</title>
<link> http://www.example.com/item1-info-page.html</link>
<description>Hebký a pohodlný svetr, ve kterém vám bude teplo i v chladnějších dnech.</description>
<g:image_link>http://www.example.com/obrazek1.jpg</g:image_link>
<g:price>25</g:price>
<g:condition>new</g:condition>
<g:id>1a</g:id>
</item>
</channel>
</rss>
```

Obrázek 4: Příklad xml feedu pro Nákupy Google

### 2.1.4 Další agregátory

Mezi další agregátory na českém internetovém poli jsou například jyxo.cz, hyperzbozi.cz, boziceny.cz. Tyto agregátory však nedosahují kvalit výše zmíněných agregátorů. Ve většině případů kombinují různé doplňkové funkce obsažené ve zboži.cz nebo heureka.cz. Jejich návštěvnost je

výrazně nižší stejně jako komfort pro zákazníky a prodejce. Z důvodu nižší návštěvnosti nejsou navíc tak atraktivní pro prodejce.

## 2.2 Agregátory slev

Agregátor slev popřípadě agregátor slevových serverů (agregátor agregátorů) je stejně jako agregátor zboží služba běžící v prostředí internetu. Jejím úkolem je zobrazování nabídek jednotlivých poskytovatelů slev popřípadě slevových serverů na jednom místě.

Slevové servery využívají faktu, že pokud službu či výrobek koupí široké spektrum lidí, klesají náklady na výrobu a prodejci či poskytovatelé služeb tak mohou poskytnout slevu. V prostředí tuzemského internetu je totiž velké množství slevových serverů (100+) a orientace mezi nimi by byla velmi nesnadná - to je hlavní důvod vzniku agregátorů slevových serverů.

Mezi nejnavštěvovanější agregátory slevových serverů patří zlateslevy.cz, slevin.cz a dealshop.cz

# 3 Využité technologie a standarty

Pro implementaci byly mimo jiné využity tyto technologie a standarty:

## 3.1 XML

XML je zkratka pro Extensible Markup Language. Jedná se o značkovací jazyk, který byl vyvinut díky potřebě jednoduchého a hlavně otevřeného formátu pro ukládání dat a informací. Data jsou zapsána v textové podobě, takže obsah je zpracovatelný v textových editorech a je nezávislý na platformě. Za vývojem a standartizací tohoto formátu stojí mezinárodní konsorcium W3C, které spravuje a vyvíjí standarty pro world wide web.

XML umožňuje tvorbu dalších značkovacích jazyků - příkladem může být XHTML. Je využíváno pro serializaci dat, pro přenos dat mezi platformami a informačními systémy.

Syntakticky by se dalo xml popsat jako strom. Každý takový dokument musí mít root element, všechny další elementy musí být uzavřeny v tomto kořenovém elementu. Všechny elementy mají počáteční a ukončovací značku. Jednotlivé elementy se nesmí překrývat. Každý element může mít libovolný počet atributů a obsah. Pokud je element prázdný je možné koncovou značku vynechat a do prvního přidat na konec tagu mezeru s lomítkem.

## 3.2 PHP

PHP je univerzální skriptovací programovací jazyk využívaný pro vývoj webových aplikací a dynamických stránek vytvořený Rasmusem Lerdorfem, Andi Gutmansem a Zeevem Suraskim . Samotný název PHP je dnes takzvanou rekurzivní zkratkou, která znamená PHP: Hypertext Preprocessor. Dříve se interpretovala jako Personal Home Page.


Obrázek 5: Logo PHP

První veřejná verze PHP byla vydaná v roce 1995. Po tomto vydání byly v druhé verzi opraveny chyby nalezené veřejností a tato verze byla označena jako PHP 2. Tato verze již měla podobnou funkčnost jako současné verze. V roce 1997 izraelští vývojáři Andi Gutmans a Zeeve Suraski přepsali parser a vytvořili tak základ pro PHP 3 a v roce 1998 bylo veřejně vydáno. V roce 1999 bylo přepsáno jádro a vydán Zend Engine. V roce 2000 bylo vydáno PHP 4 založené právě na tomto enginu. S příchodem PHP 5 roku 2004 postaveném na Zend Engine 2 přišla podpora objektově orientovaného programování. Tato verze PHP je aktuálně podporovaná a aktivně vyvíjená. Do budoucna je plánována verze 6, která by měla nativně podporovat Unicode.

PHP lze použít jak v interpretované, tak v kompilované podobě. Kompilovaná podoba se využívá pro tvorbu desktopových aplikací, kdežto interpretovaná forma je využívána pro webové prostředí. Ve webovém prostředí je PHP interpretováno na straně serveru podle aktuálních podmínek a uživatelských vstupů. Klient serveru dostane pouze odpověď přenášenou například pomocí protokolu HTTP. Formát odpovědi je v HTML nebo XHTML. Pokud je klientem mobilní zařízení, je odpověď zaslaná ve formátu WML.

Syntaxe jazyka PHP je podobná programovacím jazykům typu C a Java. PHP je platformově nezávislý a podporuje velkou škálu knihoven. Například pro tvorbu grafů, práci s XML, obrázky. Navíc je na internetu množství již napsaných tříd zdarma k dispozici. Velkou výhodou tohoto jazyka je jeho rozsáhlá dokumentace a nápověda. Obsahuje podporu mnoha databázových systémů jako MySQL, PostgreSQL, Oracle, ODBC, MSSQL. Typová kontrola jazyka PHP je slabá dynamická, což v praxi znamená, že je datový typ proměnné určený až při přiřazení hodnoty do proměnné. PHP je podporováno napříč velkým množstvím webových serveru a stalo se takřka standardem. PHP je schopno využívat funkcí operačního systému, na němž jeho interpret běží.

Za velkou výhodu tohoto skriptovacího jazyka bych označil rozsáhlé manuálové stránky s množstvím praktických příkladů.[4]

Pro jazyk PHP existuje řada aplikačních frameworků více či méně povedených. Mezi nejznáměší frameworky patří tyto:

- Zend Framework - framework podporovaný vývojáři PHP.
- Nette Framework - framework vyvíjený českými vývojáři.
- Symphony - jedná se o framework, který klade co nejvyšší nároky na úsporu psaní kódu.
- CakePHP - u tohoto frameworku je kladen důraz na jednoduchost a variabilitu.

## 3.3 MySQL

MySQL je relační databázový systém, který vytvořila firma MySQL AB. Jedná se multiplatformní databázi původně kladoucí vysoký důraz na rychlost. To však vedlo k mnohým omezením, která se postupem času stala kritickými. Tato databáze pracuje na principu klient-server, komunikace probíhá za pomoci protokolu TCP/IP a dotazovaní se provádí za pomoci jazyka SQL. V MySQL se nejvíce používají 2 enginy pro ukládání dat. Jedná se o enginy InnoDB a MyISAM. Na webu MySQL je mimo jiné dostupná dokumentace [5]


Obrázek 6: Logo MySQL

Úložiště MyISAM je nejvíce používané, mimo jiné také z důvodu, že bylo nastaveno jako výchozí až do verze MySQL 5.1. Tabulky vytvořené v tomto enginu jsou ukládány do dvou souborů. V jednom se nachází data a v druhém indexy. Jedná se o soubory .myd a . myi. MyISAM podporuje řadu funkcí, z nichž za zmínku stojí následující:

- automatické opravy
- ruční opravy
- souběh a uzamykání
- indexování - pro potřebu této bakalářské práce vyniká zejména fulltextové indexování
- pozdržené zapisování klíčů

Úložiště InnoDB je navržené pro zpracování transakcí - zejména takových, které jsou malé a časté. Toto úložiště nepodporuje fulltextové indexy, avšak na rozdíl od MyISAM podporuje od verze 3.23 již zmíněné zpracování transakcí a cizí klíče.

MySQL hlavně kvůli rychlosti nepodporovala některé funkce. Avšak v průběhu vývoje této databáze byly tyto funkce doplněny nebo budou doplněny do úložných enginů.

- verze 3.23 - InnoDB podporuje transakce a cizí klíče
- verze 4 - sjednocování dotazů pomocí UNION

- verze 4.1 - podpora funkce sounds_like (pro nalezení podobného slova - podobné jsou PHP funkce soundex, similar_text..)

- poddotazy, r-stromy(MyISAM)

- verze 5 - uložené procedury, kurzory, triggery, pohledy, information.schema
- verze 6 - do téhle verze je plánovaná podpora cizích klíčů i pro jiné enginy než InnoDB a podpora datového enginu Falcon.

## 3.4 EAN-13

EAN-13 kód slouží pro jednoznačnou identifikaci zboží.. Je tvořen 13 číslicemi. Je to číselný kód, který však může mít i podobu čárového kódu. Tento kód se skládá ze 4 částí:

- Systémový kód (kód země)
- Kód výrobce
- Kód výrobku
- Kontrolní kód

Pomocí tohoto kódu je možné také zazanmenávat ISBN kódy knih popřípadě ISSN kódy časopisů.

# 4 Možnosti systému - funkční požadavky, technické požadavky

## 4.1 Uživatelské role

V tomto systému existují 3 uživatelské role. Jedná se o neregistrovaného uživatele, registrovaného uživatele a správce systému.

**Neregistrovaný uživatel**

Neregistrovaný uživatel má právo používat agregátor zboží jako vyhledávač se všemi jeho dostupnými třídícími funkcemi.

**Registrovaný uživatel**

Registrovaný uživatel má stejná práva jako neregistrovaný, ale navíc má právo na vložení, úpravu, případně smazání adresy xml feedu, který mu patří. Může využívat funkci pro doporučení změny ceny produktu

**Administrátor systému**

Administrátor systému je uživatelská role s nejvyššími právy. Má veškerá práva co se týče všech uživatelských xml feedů. Navíc má právo potvrzovat či zamítat návrhy systému na spárování produktů.

## 4.2 Klíčové funkce

Obsahem této podkapitoly je výčet klíčových funkcí, které požadujeme po systému a jejich krátký popis.

### 4.2.1 Uživatelská sezení

Po systému požadujeme správu uživatelských sezení, korektní chování a co největší možnou bezpečnost následujících aktů:

- registrace
- přihlášení
- odhlášení

### 4.2.2 Agregátor

Po samotném jádru systému požadujeme relevantní výsledky na vyhledávací dotaz, vstup pro dotaz musí být správným způsobem ošetřen proti chybám a útokům. Dalšími požadavky jsou zobrazení detailu zboží, grafu vývoje ceny zboží a přehled obchodů, ve kterých se hledané zboží nachází.

### 4.2.3 Nastavení uživatele

Každý registrovaný uživatel systému musí mít po přihlášení možnost spravovat svá data. Jedná se zejména o následující

- přidat feed
- upravit feed
- smazat feed
- přehled feedů

### 4.2.4 Administrace

Administrátor systému musí mít k dispozici přehledy veškerých dat. Dále musí mít možnost tato data spravovat. Mezi nejdůležitější požadavky patří:

- Přehled uživatelů
- Přehled uživatelských feedů a jejich správa
- Návrhy na párování produktů

### 4.2.5 Párování produktů

Párování zboží je jednou z nejkritičtějších požadovaných funkčností. Je třeba označit stejné produkty s nestejnými názvy za identické. Po systému je vyžadováno, aby předkládal návrhy na spárování zboží, a ty pak jsou potvrzovány či zamítány administrátorem.

### 4.2.6 Doporučení změny ceny produktu

Tato funkcionalita bude dostupná pouze pro prodejce s vloženými zdrojovými feedy a povoleným přístupem k ní. Požadováno je doporučení podle uživatelských nastavení v procentech rozdílu ceny a na základě cen konkurence. Tato funkcionalita je přímo závislá na párování produktů.

## 4.3 Vstupy do systému

Do systému budou vstupovat údaje uvedené níže. Zdroji dat mohou být feedy z eshopů, stejně jako uživatelské informace následované uživatelskými rolemi a funkcemi příslušejícími těmto rolím.

- Uživatel ( heslo, kontrolní kód, email, práva, session, ip, poslední návštěva)
- Obchod ( uživatel, jméno, url, stav, url feedu, md5 feedu, smazán, ičo, název)
- Zboží ( obchod, product, productname, productnameext, description, url, imgurl, price, vat, price_vat, dues, delivery_date, shop_depots, item_type, tollfree, manufacturer, categorytext, ean, productno, date, active)
- Role ( uživatel, funkce)
- Funkce (název)

## 4.4 Výstupy ze systému

Z naimplementovaného systému požadujeme čtení následujících přehledů:

- Seznam uživatelů
- Seznam obchodů
- Seznam zboží a jejich kódů
- Graf cen
- Seznam rolí
- Seznam funkcí
- Seznam spárovaných a nespárovaných produktů
- Seznam identifikačních údajů zboží
- Výpis chyb

## 4.5 Okolí systému

S tímto systémem mohou pracovat:

- Zákazník obchodu (hledání zboží, zobrazení změn cen)
- Majitel/Administrátor obchodu (Vkládání feedů - jejich správa, sledování doporučení ohledně cen)
- Administrátor systému (Úplné řízení)

Zvláštními operátory se řídí systémové procesy. Vykonavatelem těchto procesů je aktér nazvaný SYSTÉM.

## 4.6 Technické požadavky

Po výsledném systému vyžadujeme, aby běžel na PC platformě. Provoz se předpokládá na operačních systémech Windows a UNIXových operačních systémech. Z důvodu plánovaných analýz nad získanými daty je požadavek na budoucí využití více databázových serverů.

# 5 Analýza a návrh

V této části bakalářské práce se zabývám analýzou vytvářeného systému. Jedná se o studii problému, který je třeba řešit. Celková analýza by se dala rozdělit do 2 hlavních celků – Datovou analýzu a funkční analýzu. Výstupem první z nich je struktura databáze a výstupem druhé je use case diagram a přehled funkcí se specifikacemi.

## 5.1 Datová analýza

Tato analýza se skládá z několika důležitých částí. Jedná se o lineární zápisy entit, datové slovníky a konceptuální model. Výstupem této analýzy je struktura databáze. Tato struktura databáze by měla být co nejvhodněji navržená vzhledem ke specifikaci zadání, přenositelnosti a efektivitě systému. Další podstatnou podmínkou je splnění požadavků na konzistenci, integritu a redundanci dat.

### 5.1.1 Lineární zápis entit

Tento zápis obsahuje seznam objektů – entit, které chceme evidovat v databázi. U každé entity vypisujeme její vlastnosti – atributy. Každá entita musí obsahovat klíčový – unikátní atribut, který jednoznačně identifikuje všechny ostatní atributy entity. Dále může entita obsahovat cizí atributy.
Primární (klíčový) atribut je značen podtržítkem a cizí atribut je značen tučným písmem.

**Příklad lineárního zápisu entit**

goods (**good_id,** item_id, **shop_id**, product, productname, productnameext, description, url, imgurl, price, vat, price_vat, dues, delivery_date, shop_depots, item_type, tollfree, manufacturer, categorytext, ean, productno, date, active)

Kompletní lineární zápis entit je možné nalézt v příloze.

### 5.1.2 Datový slovník

Datový slovník je soubor informací popisující strukturu tabulek v databázi. Obsahuje seznam atributů, jejich datový typ, velikost. Dále z něj vyčteme, zda může nabývat null hodnoty, jestli se jedná o klíč a jestli je indexován. Může také popisovat integritní omezení atributu. [6]

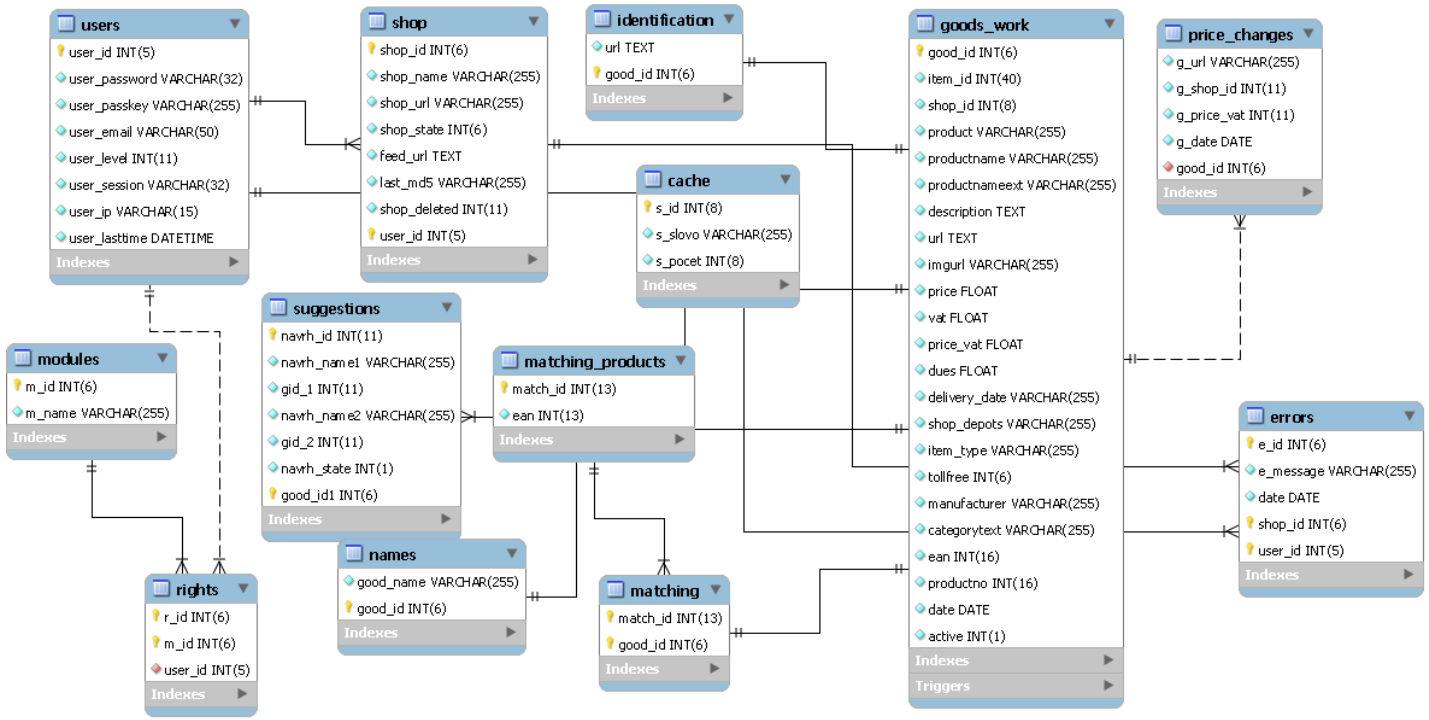
| goods | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| good_id | int | 6 | Ne | Ano |
| item_id | int | 40 | Ne | Ne |
| shop_id | int | 8 | Ne | Ano |
| product | varchar | 255 | Ano | Ne |
| productname | varchar | 255 | Ano | Ne |
| productnameext | varchar | 255 | Ano | Ne |
| description | text | | Ne | Ne |
| url | text | | Ne | Ne |
| imgurl | varchar | 255 | Ano | Ne |
| price | float | | Ano | Ne |
| vat | float | | Ano | Ne |
| price_vat | float | | Ano | Ne |
| dues | float | | Ano | Ne |
| delivery_date | varchar | 255 | Ano | Ne |
| shop_depots | varchar | 255 | Ano | Ne |
| item_type | varchar | 255 | Ano | Ne |
| tollfree | int | 6 | Ano | Ne |
| manufacturer | varchar | 255 | Ano | Ne |
| categorytext | varchar | 255 | Ano | Ne |
| ean | int | 16 | Ano | Ne |
| productno | int | 16 | Ano | Ne |
| date | date | | Ne | Ne |
| active | int | 1 | Ne | Ne |

Tabulka 1: Ukázka datového slovníku

Kompletní datový slovník je možné nalézt v příloze.

### 5.1.3 Konceptuální model

Konceptuálním modelováním můžeme popsat objekty v databázi. Zabývá se popisem tabulek, jejich atributů a vztahy mezi nimi. Výstupem tohoto modelování je databázové schéma, které je nezávislé na platformě a implementaci. Toto schéma bývá vyjádřeno ve formě ER diagramu. Na obrázku jsou zobrazeny tabulky užívané systémem. Tabulky pro testování dat a tabulky, které mají strukturu shodnou s goods_work nejsou zobrazeny.



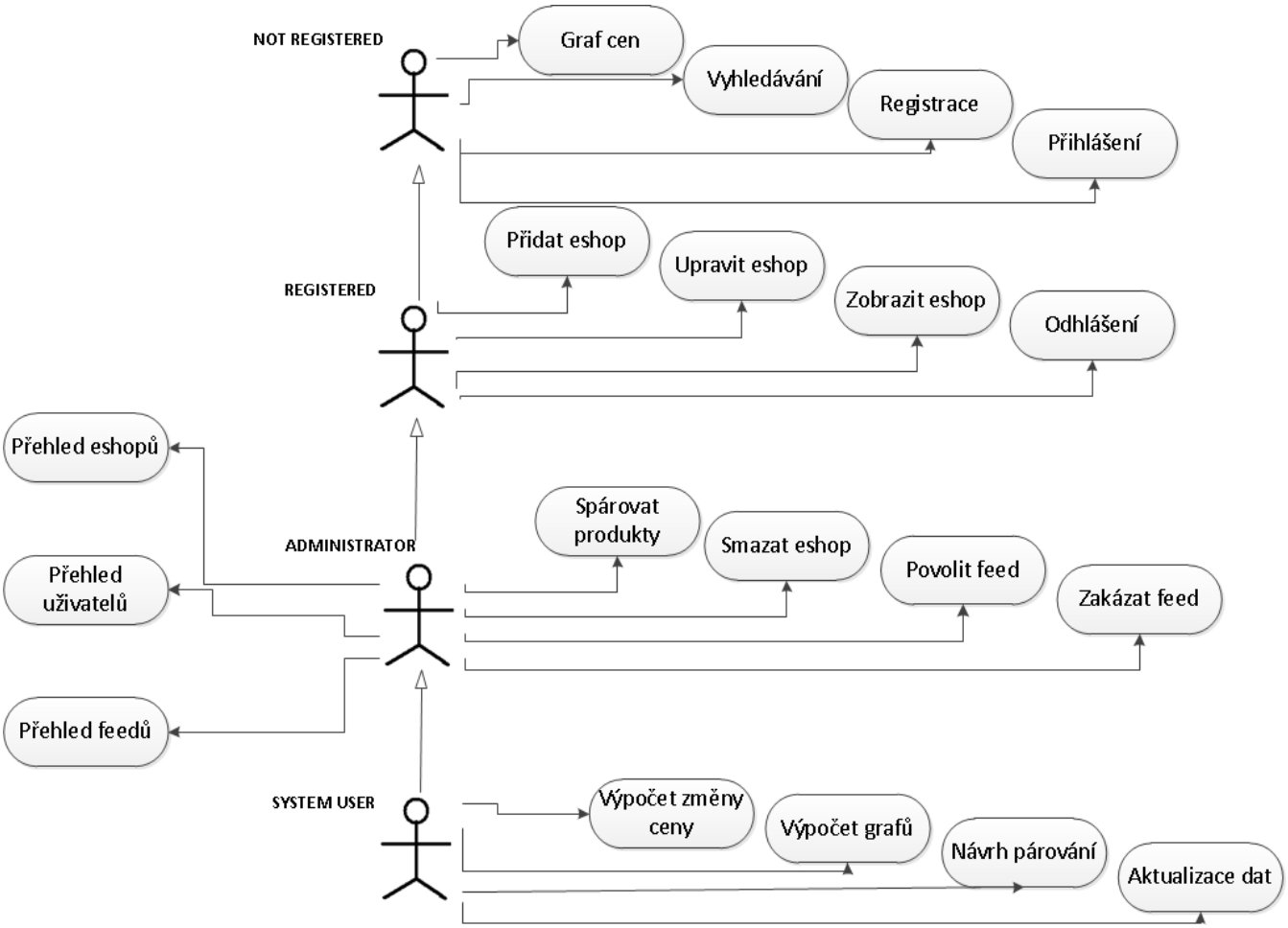

Obrázek 7: Přehled tabulek systému

## 5.2 Funkční analýza

Funkční analýza se zabývá řešením funkcí uvedených v kapitole 4 ve funkčních specifikacích. Jako první se zajímáme o use case diagram, abychom získali základní přehled o funkcích a jejich aktérech. Dále se zajímáme o specifikace funkcí a procesů. Kdo může procesy spouštět, kdo může vykonávat jaké funkce a jaké jsou vstupy a výstupy funkcí.

### 5.2.1 Use case diagram

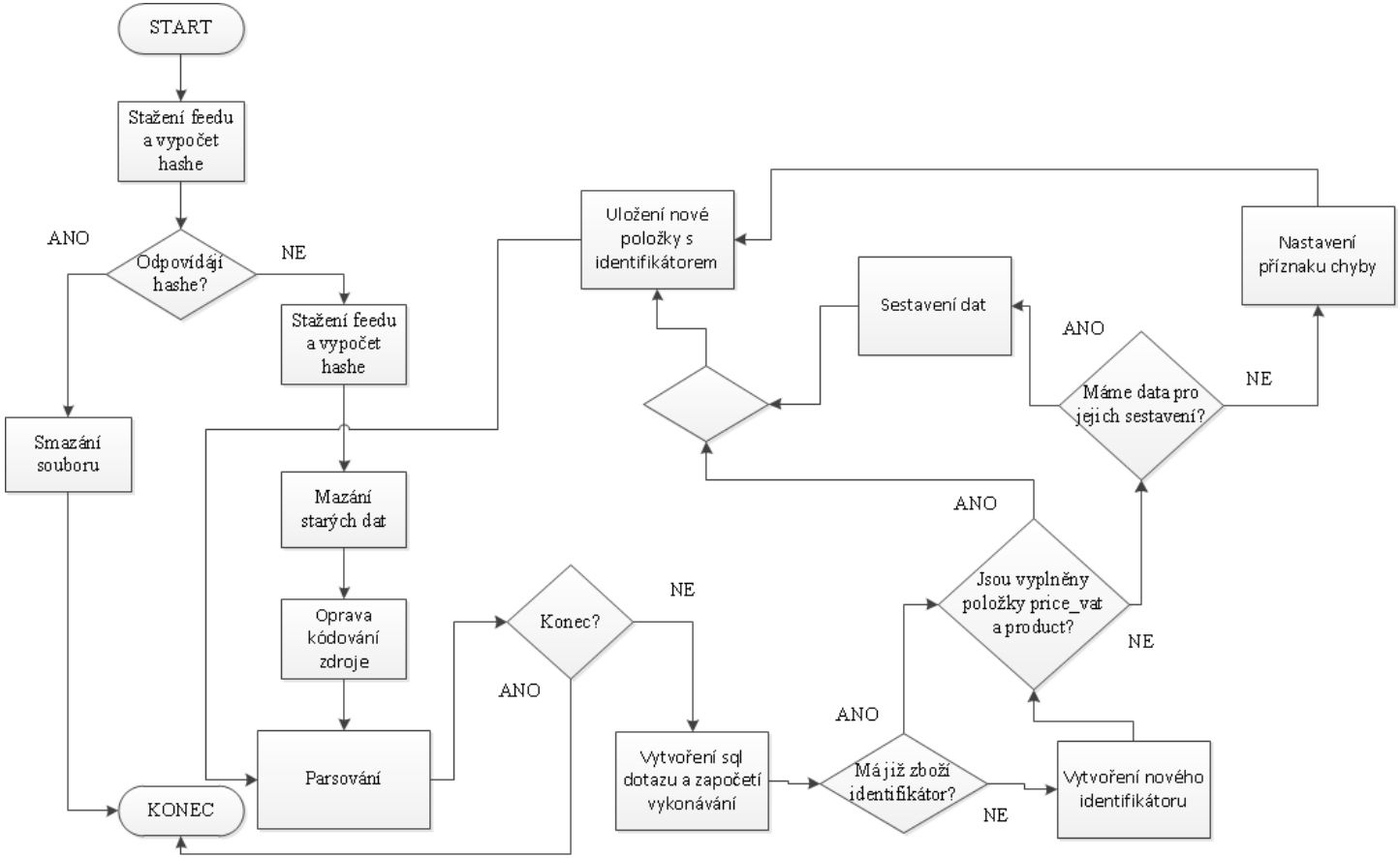
Use case diagram popisuje aktéry systému a případy užití systému. Dále popisuje vztahy mezi aktéry a vztahy mezi aktéry a případy užití. Celkově pak use case diagram znázorňuje kompletní popis funkčnosti systému. [7]




Obrázek 8: Use case diagram

### 5.2.2 Procesy a funkce

Vývojovým diagramem lze znázornit a popsat jednotlivé kroky procesu či funkce. Na následujícím diagramu je zobrazen postup jakým je do tabulky goods přidáván nový záznam.




Obrázek 9: Vývojový diagram

# 6 Implementace a testování

Po zjištění funkčních požadavků a provedení analýz již můžeme přejít k implementaci systému. Avšak ještě před zahájením samotné implementace je nutné zvážit několik faktorů a podle nich rozhodnout na jaké platformě systém poběží, jak a kde budeme ukládat data a jak bude aplikace vypadat. Poté již můžeme přejít k implementaci a testování již naprogramovaného systému.

## 6.1 Volba platformy a úložiště dat

Dnešní pokročilá doba nám nabízí velké možnosti volby programovacích jazyků, datových úložišť a dalších technologií, které je možné téměř libovolně kombinovat. Avšak je nutné přihlédnout k požadovanému prostředí, ve kterém aplikace bude spouštěna a k uživatelům, kteří ji budou využívat. Jelikož má aplikace běžet v prostředí internetu a uživatelé k ní budou přistupovat přes webové rozhraní, zvolil jsem jako programovací jazyk PHP. Kvůli výborné podpoře MySQL v PHP jsem zvolil tento SŘBD.

## 6.2 Běhové prostředí a vývojové nástroje

Před započetím implementace je třeba připravit si běhové prostředí pro spouštění a ladění systému.

**Hardwarová konfigurace**

Systém byl provozován a testován na platformě pc s následující konfigurací:

CPU: Intel Core 2 Duo CPU T6670 @ 2.20GHz
RAM: 3.00GB
GPU: ATI Mobility Radeon HD 4330

**Softwarová konfigurace**

Jako operační systém byl použit Windows Vista™ Home Premium Service Pack 2. Jako běhové prostředí jsem zvolil předpřipravené řešení v podobě programu VertrigoServ. V něm je

obsažen webový server Apache, skriptovací jazyk PHP, databázi MySQL. V tomto balíku je k dispozici také nástroj pro správu MySQL databáze PhpMyAdmin.

Implementaci systému jsem prováděl v textovém editoru PSPad, který umožňuje zvýrazňování syntaxe konkrétního jazyka.. Pro správu databáze jsem využil webového nástroje Adminer a také PhpMyAdmin.

## 6.3 Implementace

Nyní se vývoj dostává k jedné ze závěrečných etap. Je třeba využít poznatků získaných ze studia již běžících systémů, funkčních a technických požadavků a poznatků z provedených analýz. Po skončení této fáze bude existovat systém, který splňuje všechny funkční požadavky, avšak může obsahovat nějaké chyby, kvůli kterým se provádí testování.

Zároveň s implementací probíhalo testování naprogramovaných algoritmů. Důraz byl kladen na chybovost, efektivitu algoritmů a také jejich přesnost.


Obrázek 10: Úvodní stránka

## 6.4 Zjednodušené popisy zajímavých algoritmů

V této podkapitole se zabývám popisem zajímavých a klíčových algoritmů systému. Řešeny jsou převážně otázky týkající se párování zboží.

### 6.4.1 Testování uživatelských feedů

Každý zdroj dat, který je zpracovatelný algoritmem je přenesen ze zdrojového xml souboru do databáze systému, kde je testován. Toto testování je nutné k odstranění chybných dat, která se vyskytují ve zdrojových datech. Především se jedná o chyby, které by zásadním způsobem ovlivnily výsledky vyhledávání. Jsou to položky price_vat, product, description, url. V případě, že tyto položky nejsou vyplněny ani nejsou složitelné z jiných dat, je konkrétní zboží vyloučeno z vyhledávání a záznam o tom je zanesen do tabulky s chybami. Dále mohou nastat chyby spojené s nesprávně formátovaným zdrojovým souborem. Tyto problémy validity řeší knihovna Tidy.

### 6.4.2 Popis každodenní aktualizace dat pro vyhledávání:

Aktualizaci dat je nejvhodnější provádět v nočních hodinách, tedy v době menší uživatelské návštěvnosti a tudíž menšího zatížení serverů. Pro tuto aktualizaci využívám 5 tabulek. První z nich je live tabulka, nad kterou probíhá uživatelské dotazování. Tato tabulka se jmenuje goods_live.
Druhou tabulkou, do které se načítají nová nebo změněná data se jmenuje goods_work. Již neaktuální data jsou přesouvána do tabulky goods_old a na tabulce goods_fulltext jsou vytvářeny fulltextové indexy pro vyhledávání. V poslední tabulce s názvem shop, je seznam obchodů a mezi nejdůležitější sloupce patří url adresa feedu a jeho md5 otisk.

Struktura těchto tabulek byla mimo jiné navržena s ohledem na specifické požadavky českých agregátorů zboží na poskytované xml feedy.[zboží][heureka]

Využité tabulky**:**

Live tabulka, nad kterou se provádí vyhledávání - **goods_live**

Tabulka, do které se načítají nová a změněná data - **goods_work**

Tabulka se zálohami obsahující stará data - **goods_old**

Tabulka, na které se budou vytvářet indexy - **goods_fulltext**

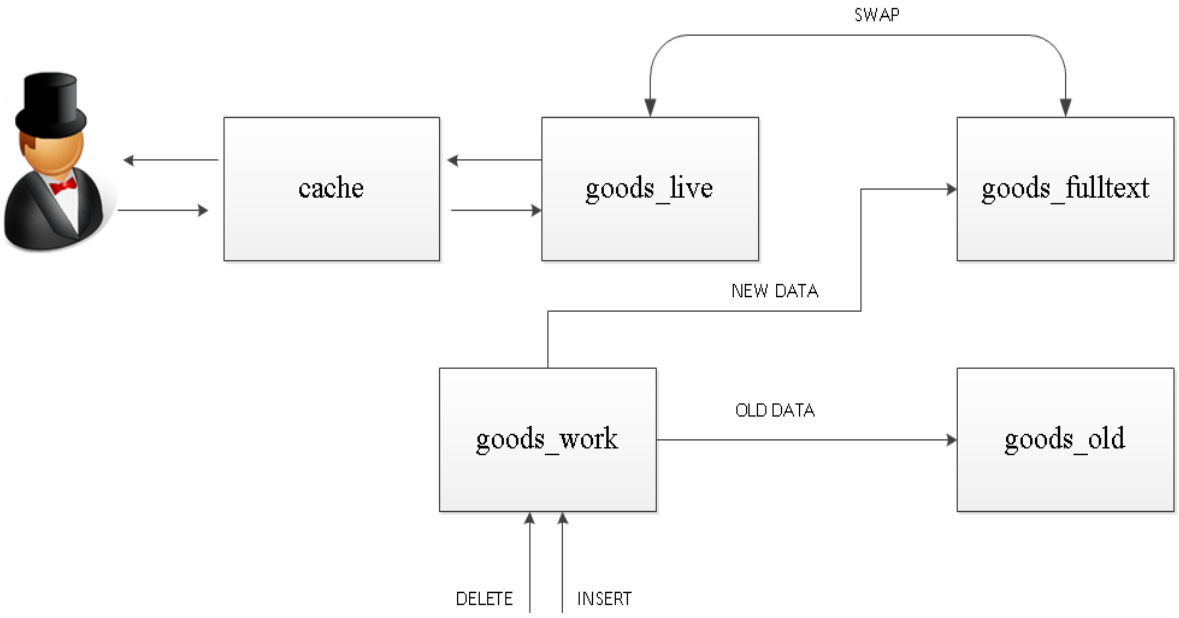
Tabulka, ve které se nachází seznam feedů - **shop**

Live tabulka goods_live obsahuje data, nad kterými se vyhledává. Indexy tabulky jsou vytvořeny nad sloupci good_id, product, price_vat. Další indexy jsou fulltextové. První je nad sloupcem product, druhý nad sloupcem description. Třetí nad oběma předchozími sloupci - tedy product a description. V tabulce goods_work se nacházejí data z předchozího dne. Obsahuje indexy good_id, price_vat a product. Tabulka goods_old obsahuje stará data a obsahuje index good_id, tabulka s názvem goods_fulltext je prázdná a s jedním indexem nad good_id.

Aktualizace dat každého obchodu probíhá následujícím způsobem:

Systém stáhne feed soubor s daty na pevný disk v počítači, vypočte jeho md5 otisk. Tento otisk porovná s otiskem uloženým v databázi. Pokud se otisky shodují, tak systém automaticky pokračuje a stahuje feed dalšího obchodu. Pokud se ovšem neshodují, tak jsou data feedu rozparsována a nahrána do databáze. Tento proces se děje se všemi feedy, které mají povolené zpracování. Po nahrátí dat do pracovní tabulky goods_work a po jejich zpracování jsou tato data systémem nakopírována do tabulky goods_fulltext, kde systém začne vytvářet fulltextové indexy pro vyhledávání, které jsou shodné s tabulkou goods_live. V případě, že by bylo k dispozici více počítačů pro zpracování by se zároveň s vytvářením indexů porovnávaly tabulky goods_live a goods_work. Rozdíly - tedy změny by se zanášely do tabulky goods_old. Avšak vzhledem k testovacímu prostředí tyto akce probíhají postupně.

Nyní se přistupuje k přejmenování tabulek. Tabulka goods_fulltext již obsahuje nová data a je třeba ji zaměnit za tabulku goods_live. Systém tedy přejmenuje goods_live na goods_temp, čímž uvolní název pro novou live tabulku a goods_fulltext na goods_live, čímž nová data s vytvořenými fulltextovými indexy zpřístupní k vyhledávání. Následuje přejmenování goods_temp na goods_fulltext, která je vyprázdněna.




Obrázek 11: Aktualizace dat

Při vkládání dat do tabulky goods_work se může stát, že nemáme k dispozici některé údaje klíčové pro fungování systému. Pokud jsou tyto údaje dohledatelné z jiných údajů jsou tyto vypočítány a dosazeny. V případě, že není vyplněn product ale jsou vyplněny položky productname a productnameext je složen product z těchto dvou údajů. Stejný proces je proveden, pokud není udaná cena (price_vat). Ta se složí z položek price a vat.

Každý produkt má v systému jednoznačný identifikátor v rámci systému - takzvané ID. To je přiřazováno podle url adresy produktu na základě předpokladu, že každý produkt má svoje url a že více produktů jej nemá stejné. V případě, že by tomu tak bylo je tento konkrétní zdroj dat vyřazen z každodenního zpracování a informace o tomto vyřazení je zanesena do tabulky chyb.
Pokud je ve feedu poskytnut údaj item_id nebo id, je uložen do databáze, v opačném případě se do sloupce tabulky uloží md5 otisk url adresy.

### 6.4.3 Nastavení MySQL pro fulltextové vyhledávání:

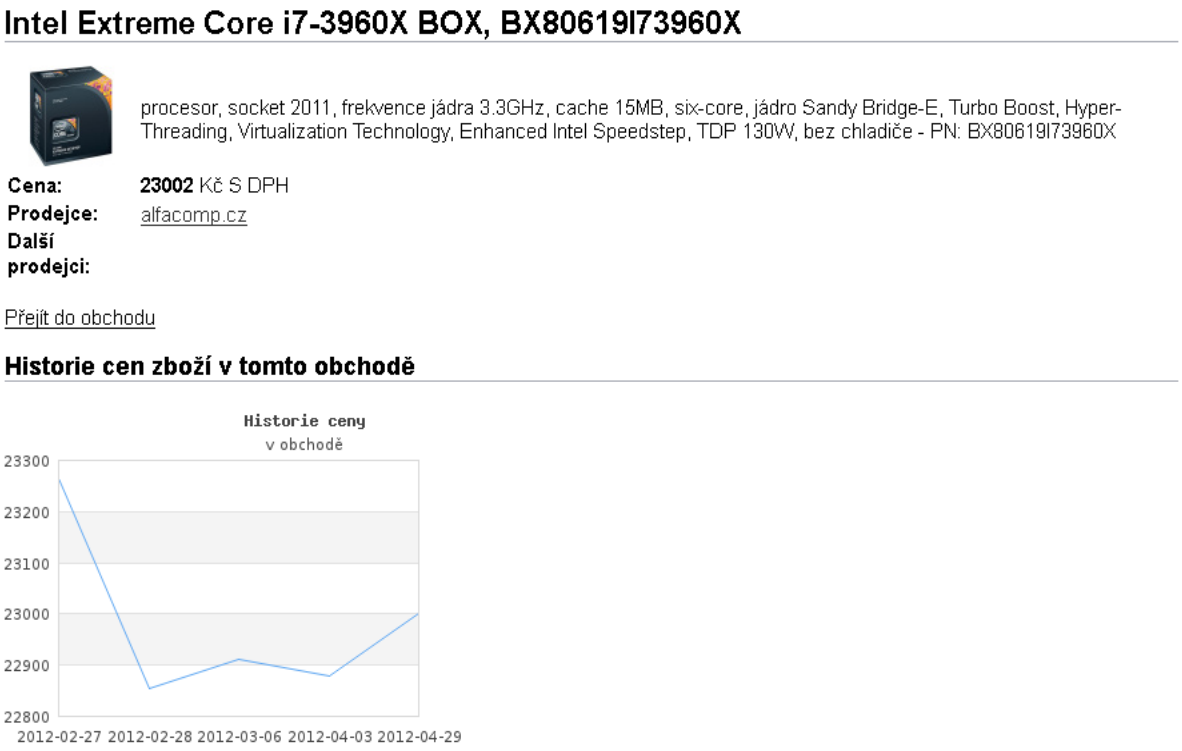
Pro správné vyhledávání bylo nutné nastavit proměnnou ft_boolean_syntax na hodnotu ' |-><()~*:\"\"&^'. To proto, aby byla vyhledávána všechna slova oddělená mezerou a ne všechna slova jako celek.

Dále bylo třeba nastavit minimální délku indexovaného slova na 2 hlavně kvůli různým řadám produktů a jejich označení. K tomuto slouží proměnná ft_min_word_len.

Přiřazením cesty k souboru words.txt do proměnné ft_stopword_file jsem docílil toho, že běžná česká slova budou při fulltextovém indexování ignorována.

### 6.4.4 Fulltextové vyhledávání

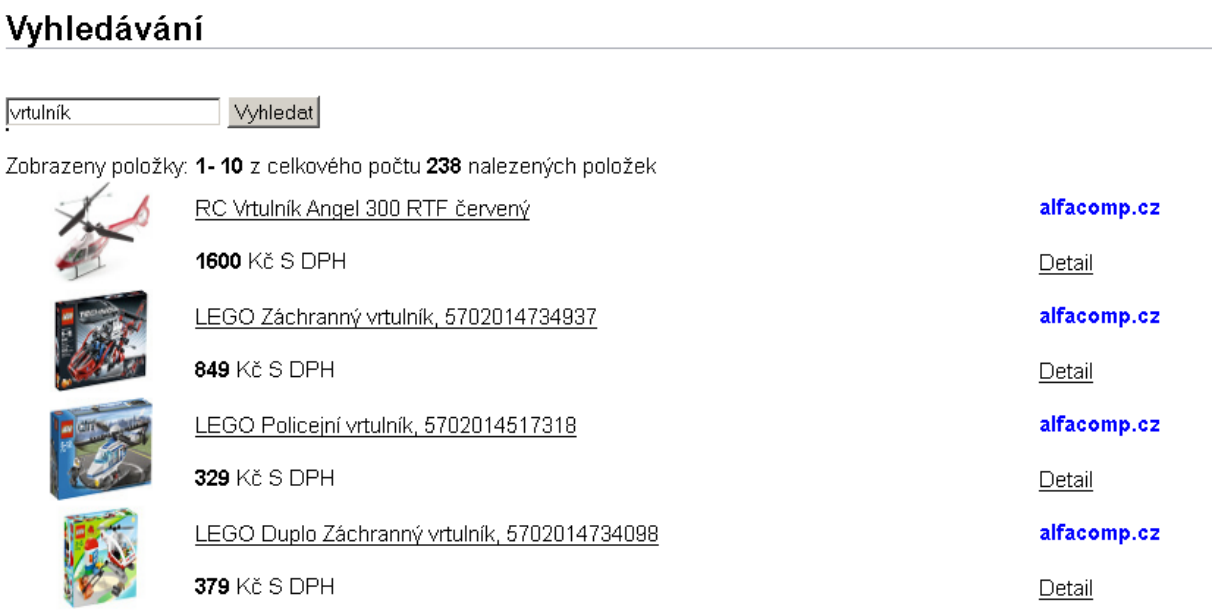
V případě, že uživatel vyhledává produkt přes agregátor, je nutné tento jeho vstup ošetřit. Hlavními důvody pro tuto činnost jsou bezpečnost a relevance vrácených výsledků. Kvůli bezpečnosti je třeba escapovat nebo převést takzvané nebezpečné znaky na znaky, které je vhodné použít. Grafy jsou generovány knihovnou JpGraph. [9]




Obrázek 12: Detail produktu po vyhledání

Nevyhovující znaky by totiž mohly vést k umožnění útoku známého jako sql injection. Pro ošetření vstupů využívám funkce mysql_real_escape_string a htmlspecialchars. První zmíněná upraví textový řetězec pro použití ve funkci mysql_query, tím že před speciální znaky vloží znak zpětného lomítka. Dále nahradí například konce řádků sekvencí znaků „\n“. Druhá funkce nahradí zvláštní znaky jejich html entitami.

Z důvodu zkvalitnění výsledků vyhledávání jsou odstraňovány přebytečné mezery a ke každému slovu je na konci přiřazen takzvaný wildcard, který částečně zajistí i skloňované varianty slova, popřípadě jeho zkratky. Mezery jsou nahrazeny znakem „+“.




Obrázek 13: Výsledky vyhledávání

### 6.4.5 Vyhledávací dotaz

Vyhledávací dotaz je tvořen dynamicky na základě uživatelských vstupů. Uživatel musí vyplnit pole, které slouží pro název zboží. Volitelně má k dispozici omezení ceny jak dolní, tak horní hranicí. Samozřejmostí je řazení zboží podle ceny.

Vyhledávací dotaz vypadá přibližně takto:

*SELECT * FROM goods_live WHERE MATCH(product, description) AGAINST ('$klic' IN BOOLEAN MODE) AND price_vat BETWEEN '$od' AND '$do' ORDER BY $priorita * MATCH(product) AGAINST ('$klic') + MATCH(description) AGAINST ('$klic') DESC*

### 6.4.6 Změna ceny produktu:

Změna ceny produktu je doporučována, pouze pokud je příslušné zboží spárováno s ostatními produkty. Z těchto produktů je vypočítána střední hodnota ceny a ta je navýšena a snížena podle uživatelského nastavení o procentuální hodnotu.

Pokud tedy střední hodnota nějakého produktu činí 2000Kč, uživatel má nastavenou odchylku 10% dostaneme hodnoty 1800Kč a 2200Kč a cena produktu v e-shopu uživatele je méně než 1800Kč popřípadě vyšší než 2200Kč, je doporučena změna ceny.

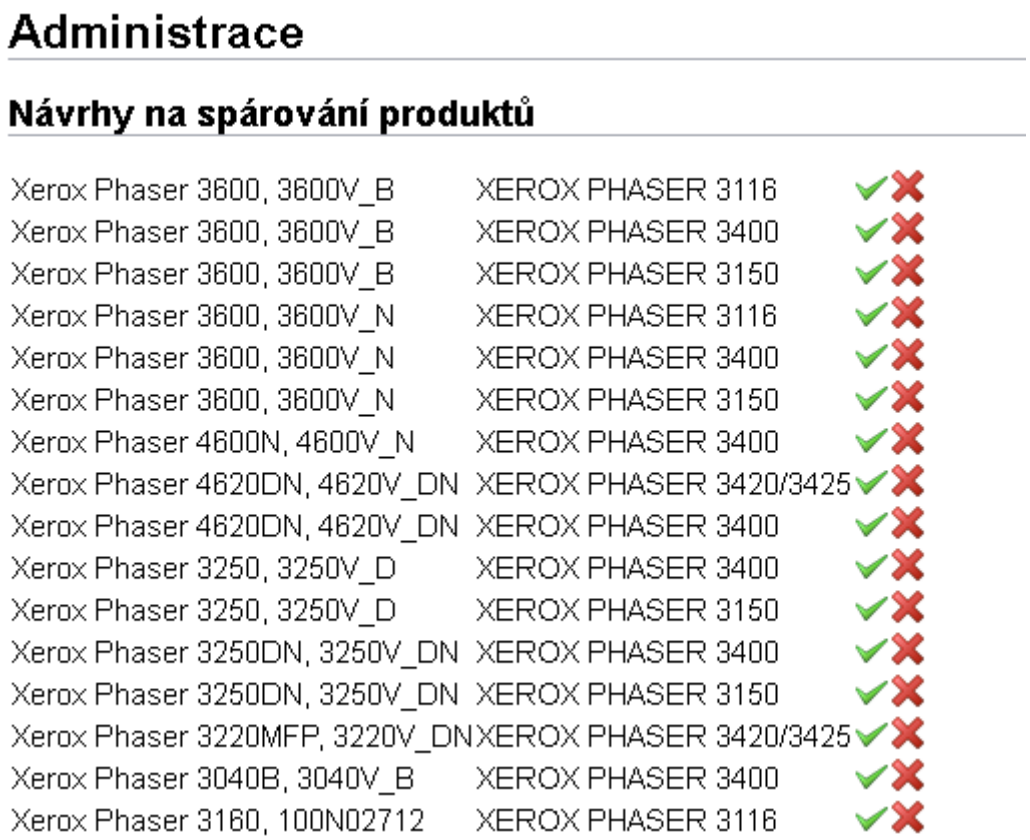
### 6.4.7 Párování produktů

Hlavním a jednoznačným kritériem pro párování produktů je EAN-13 kód, který jednoznačně identifikuje zboží. Ve většině případů, však tento kód není poskytnut. Proto jsem napsal algoritmus, který předkládá uživateli návrhy na produkty, které tento shledá totožnými, i když se jejich produktové názvy liší. Porovnával jsem různé způsoby jak na sebe stejné produkty s rozdílnými názvy napárovat.

Pro účely tohoto testování jsem předpřipravil data následujícím způsobem: Název produktu rozdělím podle mezer do pole. Odstraním netisknutelné znaky, diakritiku a jednoznakové výrazy. Velká písmena převedu na malá. Pomlčky, podtržítka ze slov kratších čtyř písmen odstraním též. Pokud v poli slov není název výrobce, tak jej tam uvedu. Poté pole seřadím dle délky slov a následně dotřídím podle abecedy. Pole převedu zpět na text, který uložím do databáze. Tato databáze textových řetězců slouží jako základ pro párování produktů.

V případě využití PHP skriptu pro párování mohou nastat 2 možnosti a to v závislosti na délce porovnávaných řetězců. Pokud jsou oba řetězce podobně dlouhé, využiji pro procentuální ohodnocení PHP funkci similar_text, která je schopna vrátit podobnost řetězců v procentech - skóre. Pokud je ovšem jeden text více než 2x delší než druhý tato funkce by i při shodném produktu, který má v názvu vícero dalších parametrů vrátila maximální procentuální podobnost 50%. Z toho důvodu jsem vymyslel druhý způsob. Ten spočívá v bodovém ohodnocení každého indexu pole podle počtu slov v poli. Pokud jsou na stejných indexech slova stejná tak jeho bodové ohodnocení přičtu k celkovému a následně přepočítám na skóre v procentech.

V případě příznivého procentuálního ohodnocení vráceného jednou z těchto metod porovnání, uložím oba názvy pod stejným identifikátorem. Toto přiřazení administrátor systému potvrzuje, případně zamítá, aby nedocházelo k párování nestejných produktů.




Obrázek 14: Návrhy na párování produktů

### 6.4.8 Metody párování produktů

**Podle EAN kódu**

Metoda párování produktů za pomoci EAN kódu je nejspolehlivější. EAN kódy jsou totiž unikátní pro každý produkt a nemůže tedy dojít k záměně produktů. Tento způsob párování produktů je také velice rychlý neboť stačí pouze vyhledat příslušný EAN kód v zaindexovaném sloupci tabulky. Avšak tento kód je ve většině případů neznámý, proto není možné využít jej pro párování.

**V PHP skriptu pomocí funkce similar_text**

Funkce similar_text vrací procentuální podobnost dvou textových řetězců. Tato funkce má kubickou složitost tedy $O(N^3)$, takže pokud se velikost dat zdvojnásobí čas potřebný pro vykonání

funkce je osmkrát delší. Pro delší názvy produktů je tedy tato funkce časově náročná. Jako další nevýhoda by se dal označit fakt, že pokud máme například 2 řetězce o délkách 10 a 20 znaků, tak bude maximální procentuální podobnost 50% v případě, že první řetězec bude podřetězcem druhého. Tato možnost by nastala, kdyby dva shodné produkty byly uvedeny jako jeden bez parametrů a druhý s parametry. Tedy v případě kdy jsou 2 řetězce takto rozdílné, není možné použít tuto metodu.

**V php skriptu ohodnocením indexů pole**

Název produktu je rozdělen podle mezer do pole a seřazen dle abecedy popřípadě podle délky slov a následně podle abecedy. Každý index pole je bodově ohodnocen. Nultý index má ohodnocení 100 bodů, každý další index má hodnocení 0.75x menší. Ohodnocovací pole má stejný počet prvků jako pole produktu s delším názvem. Řetězce na stejných indexech pole jsou porovnány, a pokud jsou stejné tak se přičtou k výslednému skóre. Algoritmus tak zvýhodňuje delší slova, popřípadě slova, která jsou na začátku abecedy (čísla), kterým přiděluje vyšší ohodnocení. Pro velký počet porovnávaných názvů produktů je tato metoda velmi pomalá.

**Na databázové vrstvě fulltextem**

Při fulltextovém vyhledávání je vyhledáván výraz proti tabulce upravených názvů produktů. Jako relevantní jsou brány výsledky, u kterých použitá klauzule MATCH-AGAINST vrátila skóre větší než 9. Tato metoda je 3,187x rychlejší než použití dotazu LIKE, nutno však podotknout, že tato hodnota notně závisí na konkrétním vzorku dat.

Ukázka části procedury pro párování:

```
SELECT s_id, s_slovo FROM slova_test;
SELECT product_match_search, MATCH (`product_match_search`) AGAINST (s_slovox) as skore
FROM test WHERE MATCH (`product_match_search`) AGAINST (s_slovox) ORDER BY skore;
INSERT INTO results (pms, dotaz, zpusob) VALUES (pms,s_slovox,0);
```

**Na databázové vrstvě za pomoci LIKE**

Stejně jako u předchozího způsobu se vyhledává proti tabulce upravených názvů produktů výrazem ve tvaru "%hledany%vyraz%". V malém procentu případů je tato metoda přesnější než fulltextové vyhledávání, ale díky vysoké režii se nevyplatí. Tento způsob je výrazně - 3x - pomalejší než fulltextové vyhledávání.

Ukázka části procedury pro párování:

*SELECT s_id, s_slovo FROM slova_test;*
*SELECT product_match_search FROM test WHERE product_match_search LIKE like_dotaz;*
*INSERT INTO results (pms, dotaz, zpusob) VALUES(pms, s_slovox, 1);*

# 7 Časové a výpočetní náklady systému

V této kapitole se zabývám výpočetními nároky naimplementovaného systému zejména z časového hlediska.

## 7.1 Nejnáročnější procesy systému

Během testování bylo vypozorováno, že nejnáročnějším procesem v systému z časového hlediska je aktualizace dat. Pro měření času při aktualizaci dat byl použit vzorek dat obsahující přibližně 19200 záznamů o velikosti přibližně 11,7MB.

### 7.1.1 Každodenní aktualizace dat

Každodenní aktualizace dat se skládá z procesů uvedených níže. Čas A ukazuje čas potřebný pro vykonání procesu, když je databáze prázdná. Časy B a C se rozumí čas, kdy jsou již v tabulkách nějaká data.

**Stažení XML feedu**

Doba stahování souboru xml feedu je závislá na mnoha faktorech. Předpokládejme konstantní rychlost přenosu každého feedu – tedy, že hardware, na kterém běží systém, je schopen přijímat data pomalejší rychlostí, než je kterýkoli server, ze kterého stahujeme feed schopen tato data poskytovat Tímto předpokladem jsme eliminovali faktory pouze na jeden a to velikost stahovaného souboru.

**Parsování a vkládání do databáze**

Ještě před samotným zpracováním dat, je třeba zajistit správnou syntaxi xml souboru, a pokud není v kódování UTF-8 jeho překódování. Poté již následuje samotné rozparsování. Sestaví se sql dotaz z ošetřených dat a následně se provede. Pro co největší úsporu času je co nejvíce aplikační logiky prováděno na databázové vrstvě.

**Překopírování nových dat**

V okamžiku, kdy máme k dispozici nová data v pracovní tabulce, jsou tato data překopírována do tabulky, kde se vytvoří indexy. Měřen je pouze čas kopírování a po tomto procesu máme k dispozici 2 tabulky se shodnými daty.

**Vytvoření indexů**

Po překopírování dat je třeba vytvořit indexy. Jedná se jak o fulltextové indexy určené pro vyhledávání, tak o indexy, které budou potřeba pro identifikaci a zálohu již neaktuálních dat.

**Identifikace a záloha starých dat**

Porovnáním live tabulky s pracovní tabulkou za pomocí SQL klauzule LEFT OUTTER JOIN identifikuji stará data, a ta přesunu do tabulky se zálohami.

**Údržba tabulek**

Tento proces zahrnuje prohození live tabulky a tabulky s novými daty, na které jsou vytvořeny indexy. Dále probíhá mazání nepotřebných dat a indexů.

**Generování dat pro grafy**

Pod tímto procesem se skrývá přesunutí části dat z tabulky záloh do tabulky změn cen.

V následující tabulce jsou zobrazeny časové údaje o aktualizaci.

| | **Nová data [s]** | **Aktualizace 1 [s]** | **Aktualizace 2 [s]** |
|---|---|---|---|
| **Stažení XML feedu** | 4.0 | 4.1 | 3.9 |
| **Parsování a vkládání do databáze** | 256.8 | 488.6 | 500.7 |
| **Překopírování nových dat** | 0.2 | 0.3 | 0.2 |
| **Vytvoření indexů** | 44.7 | 45.7 | 47.0 |
| **Identifikace a záloha starých dat** | 0 | 0.9 | 0.9 |
| **Údržba tabulek** | 0.2 | 0.3 | 0.4 |
| **Generování dat pro grafy** | 0 | 1.3 | 1.3 |
| **Celkový čas aktualizace** | 306.3 | 544.7 | 559.8 |

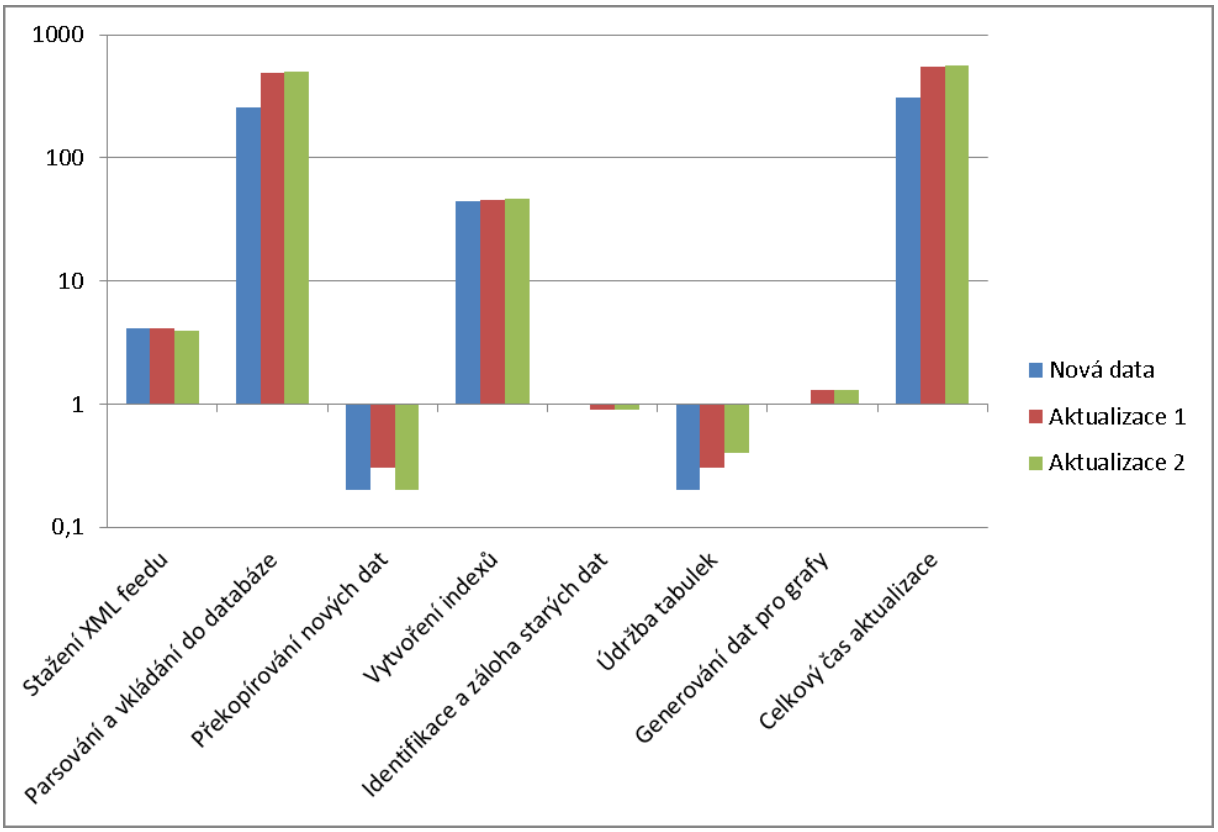
Tabulka 2: Časové přehledy funkcí

**Celkový čas aktualizace**

Celkový čas aktualizace zahrnuje všechny procedury spojenou s aktualizací. Je vidět, že počáteční naplnění daty je celkově rychlejší než aktualizace. V průběhu aktualizace se totiž musí vypočítávat jednoznačné identifikátory položek zboží v systému.

### 7.1.2 Předpokládaná časová náročnost reálně nasazeného systému

Otázka časové náročnosti zpracování dat výsledného systému, který je naplněn daty, je velmi důležitá. Je vhodné odhadnout tuto náročnost a zjistit tak jak dlouho by operace aktualizace trvala.




Obrázek 15: Grafické znázornění časových režií aktualizace

České agregátory zboží dnes indexují přibližně 4 až 5 milionů různých druhů zboží. Při uvážení tohoto faktu a prostým vynásobením počtu zpracovávaných položek se dostáváme na číslo 23 představující počet hodin pro jednorázové vložení těchto položek do systému. V případě aktualizace všech položek bychom se dostali na 39 hodin. Tyto časové hodnoty jsou pouze teoretické, avšak alarmující.

### 7.1.3 Statistika nahrávání dat do systému

Pro účely tohoto testování byly náhodně vybrány 4 obchody. Z těchto čtyř obchodů byly staženy jejich xml feedy a data z nich byla naimportována do systému. V následující tabulce jsou

časové přehledy tohoto testování. Zobrazen je celkový i průměrný čas stahování feedu a celkový i průměrný čas potřebný pro jeho zpracování. Stejným způsobem je zobrazen počet položek a počet chybných položek.

| | Celkem | Průměrně |
|---|---|---|
| Čas stahování [s] | 11 | 2,75 |
| Celkový čas [s] | 357 | 89,25 |
| Počet položek | 22238 | 5559,5 |
| Chybných položek | 13 | 3,25 |

Tabulka 3: Statistika nahrávání dat

### 7.1.4 Faktory ovlivňující rychlost aktualizace

Faktorů, které ovlivňují rychlost aktualizace, je celá řada. Zejména se jedná o aktuální využití systémových zdrojů počítačové sestavy, na které systém běží, vytíženosti internetového připojení atd.

V případě uvedeném výše se počítalo pouze s unikátními položkami. Avšak je více než pravděpodobné, že stejná položka je ve více internetových obchodech současně. S tímto faktem tedy narůstá objem dat ke zpracování. Z opačného hlediska se ovšem dá předpokládat, že alespoň jedna třetina internetových obchodů aktualizuje průměrně 1x týdně, další třetina dvakrát až třikrát týdně a poslední skupina e-shopů aktualizuje svoje feedy denně.

Z předchozího odstavce ovšem není patrné, jaká vlastně bude časová náročnost na aktualizaci. Proces aktualizace je závislý na velkém množství vnitřních i vnějších okolností, které v tuto chvíli není možné předpovídat. Tyto údaje budou k dispozici až po reálném spuštění systému.

# 8 Výhody ukládání uživatelských dotazů

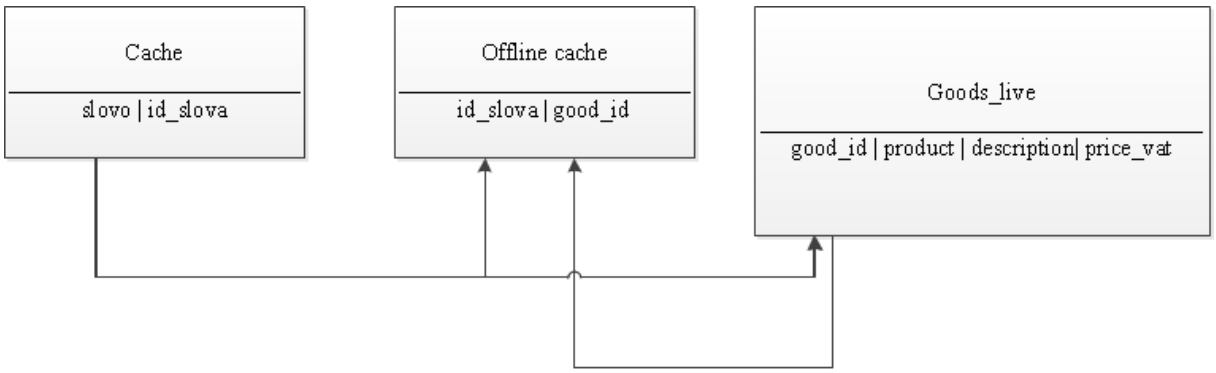
Základním předpokladem pro využití uživatelských dotazů je, že při každém uživatelském hledání uložíme uživatelský dotaz do databáze. Pokud byl dotaz již položen, tak se pouze inkrementuje jeho počítadlo. Tato databáze uživatelských dotazů má mnohá využití. V první řadě je možné z tohoto seznamu při spuštění systému předgenerovat výsledky vyhledávání do cache tabulky, která se nachází v datovém úložišti MySQL typu MEMORY. Toto datové úložiště má tu nespornou výhodu, že se nachází v paměti RAM a tak umožňuje velice rychlé vracení výsledků. Jako další možnost využití se jeví automatické opravy překlepů uživatelů. A v neposlední řadě by bylo možné provádět analýzy nad těmito dotazy a určovat například popularitu konkrétního zboží.

## 8.1 Předgenerování výsledků do cache

Pokud uživatel dostane odpověď na svůj dotaz z cache systému – jedná se z našeho pohledu o takzvaný laciný dotaz.[8]. Jak již bylo zmíněno, je tato cache generována ze zadaných uživatelských dotazů. Generování těchto výsledků probíhá denně po každé aktualizaci dat, aby byla zajištěna integrita a relevance výsledků.

### 8.1.1 Postup generování

Z tabulky cache postupně vyberou všechny záznamy a tyto jsou fulltextově vyhledány v live tabulce. Identifikátory získané z fulltextového hledání jsou uloženy do tabulky offline_cache, kde jsou jim zpětně přiřazeny identifikátory z tabulky cache.

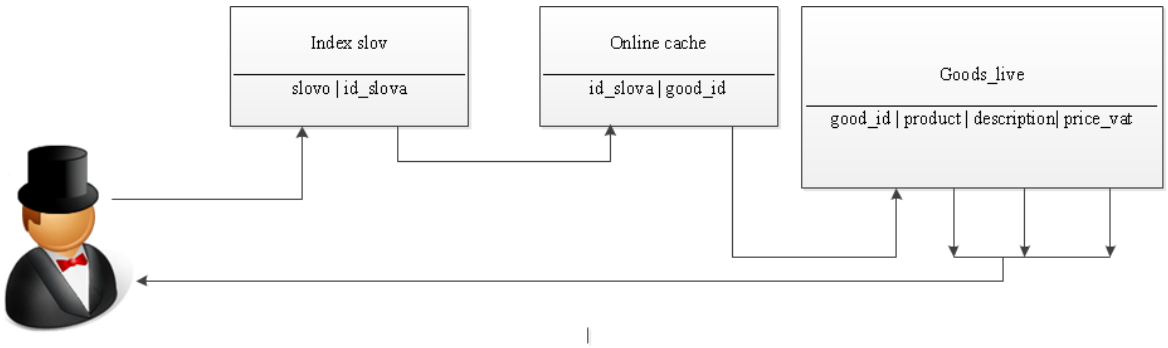



Obrázek 16: Generování cache

Následně jsou v této tabulce vytvořeny indexy nad sloupcem id_slova. Ve vhodém okamžiku jsou zaměněny a přejmenovány tabulky offline_cache a online_cache.

### 8.1.2 Postup vrácení výsledku

Uživatelský dotaz je vyhledáván v tabulce index_slov. Pokud tam není nalezen, tak systém přejde ke klasickému fulltextovému vyhledávání v live tabulce. Pokud je ovšem záznam nalezen, tak se vybere id dotazu, ke kterému je následně přiřazen z tabulky online_cache seznam identifikátorů zboží z live tabulky. Následuje již jen vyhledání těchto identifikátorů v zaindexovaném sloupci live tabulky a uživateli jsou vráceny výsledky.




Obrázek 17: Vrácení výsledků z cache

## 8.2 Opravy překlepů

V případě, že uživatel zadá do vstupu vyhledávání nekorektní slovo – tedy se uživatel při zadávání překlepl je z části možné tento jev překlenout. K tomu je vhodné využít PHP funkci similar_text, která již byla zmíněná v metodách párování produktů. [10]

Postup při opravách překlepů je následující. Pokud se uživatel takzvaně překlepne a z cache nebo z fulltextového vyhledávání nejsou vráceny žádné výsledky, tak je za pomoci funkce similar_text nalezen z tabulky cache záznam, který nejvíce odpovídá zadanému dotazu. Tento dotaz je již cachován a proto jsou okamžitě vráceny předgenerované výsledky z paměti RAM.

# 9 Zhodnocení výsledků a srovnání s konkurencí

Naimplementovaný systém umožňuje zákazníkům internetových obchodů vyhledat žádané zboží a následně jej filtrovat a řadit dle zadaných kritérií. Po vyhledání konkrétního zboží má uživatel možnost přejít na kartu produktu, kde nalezne jeho detailní popis, cenu, prodejce konkrétního zboží, které vyhledal a seznam dalších prodejců stejného zboží. Dále je na této kartě zobrazen graf vývoje ceny daného produktu v obchodě.

Pro obchodníky nabízí systém kompletní přehled a správu jejich obchodů. Dále má obchodník možnost vyhledat své zboží a zjistit jak si stojí v porovnání s konkurencí. Systém mu za předpokladu korektních dat a spárování zboží nabízí doporučenou cenu.

Mezi přednosti tohoto systému bezpochyby patří funkce, která doporučí prodejní cenu a graf vývoje cen. První z nich nabízí potenciální výhodu uživateli systému oproti konkurenčním prodejcům. Graf vývoje cen, může na druhou stranu sloužit jako základ pro predikci dalšího vývoje ceny, zejména pro zákazníka elektronických obchodů

Za slabinou tohoto řešení by se dalo označit párování zboží. I přes mnohá testování různých algoritmů se nepodařilo najít vhodný způsob jak jej realizovat efektivněji nebo alespoň stejně dobře jako konkurenční systémy.

Dalším problémem se jeví vysoké časové režie. Ty by se daly řešit částečnou paralelizací jak v databázové části, tak v části programu. Další možností je změna platformy popřípadě programovacího jazyka, který má pro tyto konkrétní úlohy lepší předpoklady. Za zmínku stojí také to, že PHP je k dispozici také v kompilované verzi a nevznikla by tak potřeba systém přepisovat.

Veškeré chování systému by bylo vhodné otestovat v reálném provozu. Avšak ještě předtím je třeba optimalizovat algoritmus párování zboží a časové režie. Poté již reálnému nasazení nic nebrání a nad získanými daty z provedených testů by se dal ladit systém podle konkrétních požadavků dnešního internetového světa.

Nad získanými daty z reálného provozu by bylo následně také možné provádět jiné analýzy týkající se spíše obchodní stránky věci. Namátkou by se daly například vybrat roční analýzy změn cen zboží. Stará a neaktuální data přemisťovat do datových skladů pro další analytické dotazování.

# 10 Závěr

Tato práce byla zaměřena na seznámení se s problematikou agregátorů zboží. Nastudování funkcí těchto agregátorů, odhad procesů probíhajících v jejich systémech a navrhnutí vlastního systému na základě těchto poznatků s přidanou funkcionalitou. Výsledkem je agregátor zboží s párováním zboží a funkcí, která doporučí změnu ceny produktu, na základě konkurence.

V první části této práce jsem popsal tuzemskou situaci na poli agregátorů zboží a jejich funkcionalitu. Dále jsem se zabýval popisem technologií a standardů, které se využívají v těchto systémech a které byly využity pro implementaci vlastního systému.

Byly specifikovány funkční a technické požadavky a byla provedena datová a funkční analýza. Díky poznatkům získaným z předchozích činností jsem naimplementoval výsledný systém. Při jeho vývoji byl kladen důraz především na co nejvyšší efektivitu a přesnost na úkor uživatelské přívětivosti.

Nyní je datové úložiště systému prázdné, neboť není k dispozici žádný svobodný zdroj dat, ze kterého by bylo možné tato data získat. Takže dalším logickým krokem při vývoji tohoto systému by bylo reálné nasazení a odladění chyb zjištěných reálným provozem. Pro zrychlení vyhledávání navíc zkusit implementovat vlastní indexování. Prací do budoucna také zůstává informovat potenciální uživatele systému a zákazníky, kteří budou tento systém využívat například reklamní kampaní na internetu s poukázáním na přidané hodnoty systému oproti konkurenci.

# Seznam použité literatury


[1] Specifikace XML souboru. *Služby obchodům - Heureka.cz* [online]. 2012 [cit. 2012-05-02]. Dostupné z: http://sluzby.heureka.cz/napoveda/xml-feed/

[2] Specifikace XML pro internetové obchody. *Zboží.cz* [online]. 2012 [cit. 2012-05-02]. Dostupné z: http://napoveda.seznam.cz/cz/zbozi/napoveda-pro-internetove-obchody/specifikace-xml/

[3] Specifikace zdroje produktů. In: *Nákupy google* [online]. 2012 [cit. 2012-05-02]. Dostupné z: http://support.google.com/merchants/bin/answer.py?hl=cs&answer=188494&ctx=cb&src=cb&cbid=ths040oqy95x#other

[4] PHP Manual. *PHP* [online]. 2012 [cit. 2012-05-02]. Dostupné z: http://www.php.net/manual/en/index.php

[5] MySQL Documentation: MySQL Reference Manuals. *MySQL* [online]. 2012 [cit. 2012-05-02]. Dostupné z: http://dev.mysql.com/doc/

[6] GÁLA, Libor, Jan POUR a Zuzana ŠEDIVÁ. *Podniková informatika*. 2., přeprac. a aktualiz. vyd. Praha: Grada, 2009, 496 s. Expert (Grada). ISBN 978-80-247-2615-1.

[7] ZELINKA, Tomáš a Miroslav SVÍTEK. *Telekomunikační řešení pro informační systémy síťových odvětví*. 1. vyd. Praha: Grada, 2009. Organizace UML modelu, s. 56.

[8] Happy hours – uvolnění limitů hledání. In: *Blog fulltextového týmu* [online]. 2012 [cit. 2012-05-02]. Dostupné z: http://fulltext.sblog.cz/2012/04/26/happy-hours-uvolneni-limitu-hledani/

[9] *JpGraph - Most powerful PHP-driven charts* [online]. 2012 [cit. 2012-05-02]. Dostupné z: http://jpgraph.net/

[10] VRÁNA, Jakub. Překlepy ve vyhledávání. In: *PHP triky* [online]. 2007 [cit. 2012-05-02]. Dostupné z: http://php.vrana.cz/preklepy-ve-vyhledavani.php

# Přílohy

**Příloha A. Datový slovník**

| errors | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| e_id | int | 6 | Ne | Ano |
| u_id | int | 6 | Ano | Ne |
| s_id | int | 6 | Ano | Ne |
| e_message | varchar | 255 | Ne | Ne |
| date | date | | Ne | Ne |

| goods | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| good_id | int | 6 | Ne | Ano |
| item_id | int | 40 | Ne | Ne |
| shop_id | int | 8 | Ne | Ano |
| product | varchar | 255 | Ano | Ne |
| productname | varchar | 255 | Ano | Ne |
| productnameext | varchar | 255 | Ano | Ne |
| description | text | | Ne | Ne |
| url | text | | Ne | Ne |
| imgurl | varchar | 255 | Ano | Ne |
| price | float | | Ano | Ne |
| vat | float | | Ano | Ne |
| price_vat | float | | Ano | Ne |
| dues | float | | Ano | Ne |
| delivery_date | varchar | 255 | Ano | Ne |
| shop_depots | varchar | 255 | Ano | Ne |
| item_type | varchar | 255 | Ano | Ne |
| tollfree | int | 6 | Ano | Ne |
| manufacturer | varchar | 255 | Ano | Ne |
| categorytext | varchar | 255 | Ano | Ne |
| ean | int | 16 | Ano | Ne |

| | | | | |
|---|---|---|---|---|
| productno | int | 16 | Ano | Ne |
| date | date | | Ne | Ne |
| active | int | 1 | Ne | Ne |

| identification | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| good_id | int | 10 | Ne | Ano |
| url | text | | Ne | Ano |

| matching | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| match_id | int | 13 | Ne | Ne |
| good_id | int | 13 | Ne | Ne |

| matching products | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| match_id | int | 13 | Ne | Ano |
| ean | int | 13 | Ne | Ne |

| modules | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| m_id | int | 6 | Ne | Ano |
| m_name | varchar | 255 | Ne | Ne |

| names | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| good_id | int | 13 | Ne | Ano |
| good_name | varchar | 255 | Ne | Ne |

| price_changes | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| g_url | varchar | 255 | Ne | Ano |
| g_shop_id | int | 11 | Ne | Ne |
| g_price_vat | int | 11 | Ne | Ne |
| g_date | date | | Ne | Ne |
| g_id | int | 11 | Ne | Ano |

| rights | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| r_id | int | 6 | Ne | Ano |
| user_id | int | 6 | Ne | Ne |
| m_id | int | 6 | Ne | Ne |

| shop | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| shop_id | int | 6 | Ne | Ano |
| user_id | int | 6 | Ne | Ne |
| shop_name | varchar | 255 | Ne | Ne |
| shop_url | varchar | 255 | Ne | Ne |
| shop_state | int | 6 | Ne | Ne |
| feed_url | text | | Ne | Ne |
| last_md5 | varchar | 255 | Ne | Ne |
| shop_deleted | int | 1 | Ne | Ne |

| suggestions | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| navrh_id | int | 13 | Ne | Ano |
| navrh_name1 | varchar | 255 | Ne | Ne |
| gid_1 | int | 13 | Ne | Ne |
| navrh_name2 | varchar | 255 | Ne | Ne |
| gid_2 | int | 13 | Ne | Ne |
| navrh_state | int | 1 | Ne | Ne |

| users | | | | |
|---|---|---|---|---|
| Atribut | Datový typ | Velikost | Null | Klíč |
| user_id | int | 5 | Ne | Ano |
| user_password | varchar | 32 | Ne | Ne |
| user_passkey | varchar | 255 | Ne | Ne |
| user_email | varchar | 50 | Ne | Ne |
| user_level | int | 11 | Ne | Ne |
| user_session | varchar | 32 | Ne | Ne |

| | | | | |
|---|---|---|---|---|
| user_ip | varchar | 15 | Ne | Ne |
| user_lasttime | datetime | | Ne | Ne |

**Příloha B. Lineární zápis entit**

errors(e_id, **u_id**, **s_id**, e_message, date)
goods (**good_id,** item_id, **shop_id**, product, productname, productnameext, description, url, imgurl, price, vat, price_vat, dues, delivery_date, shop_depots, item_type, tollfree, manufacturer, categorytext, ean, productno, date, active)
identificatio(**good_id, url**)
matching(**match_id, good_id**)
matching_products(match_id, ean)
modules(m_id, m_name)
names(**good_id**, good_name)
price_changes(**g_url, g_shop_id, g_price_vat.** g_date**, g_id**)
rights(r_id, **user_id, m_id**)
shop(shop_id, **user_id**, shop_name, shop_url, shop_state, feed_url, last_md5, shop_deleted)
suggestions(navrh_id, **navrh_name1, gid_1, navrh_name2, gid_2**, navrh_state)
users(user_id, user_password, user_passkey, user_email, user_level, user_session, user_ip, user_lasttime)